

Written Exam

Part 1 – Quizz (40%)

You should answer 8 out of the following 10 questions.

1. What is a transaction? You should define the concept of transaction in a sentence (or two).

A transaction is a group of operations for which the DBMS guarantees some properties.

2. What does C stand for in ACID transaction properties? Name and define this concept in a few sentences.

C stands for consistency. It means that a transaction brings the database from a consistent state to another consistent state. A consistent state is defined by the constraints of the database model.

3. What is a foreign key?

A foreign key is an attribute from a relation R which references a primary key in another relation S.

4. Assume relation R(A, B, C) is in BCNF and AB is key. Can a functional dependency A → C be valid?

No, in BCNF the only valid FDs are the ones where the left hand side is the key.

5. Express the relational division operator using basic relational operators (set operations, project, select, cardinal product, renaming)

a division is obtained with a double negation (assuming the appropriate schemas where the attributes of s that are also attributes of r are designed as B, and the attributes of r which are not in s as A):

All the attributes A of r minus those attributes that are not associated to all B attributes.

$r / s = \text{project}_A(r) - \text{project}_A((\text{project}_A(r) \times s) - r)$

6. Express the following SQL query in relational algebra (assume that schema are union compatible as required):

SELECT r.a FROM r WHERE r.b not in (select s.b from s);

$rr = \text{project}_b(r) - \text{project}_b(s)$

$s = \text{select}_{rr.b=r.b}(rr \times r)$

$\text{project}_a(s)$

7. Express the following SQL query in relational algebra (assume that schema are union compatible as required):

```
SELECT * FROM r EXCEPT SELECT * from s;
```

$r - s$

8. How is it possible to represent a tertiary relationship (a relationship between three entities) in UML using only binary associations?

Create a new entity class and create binary associations between the new class and the three classes that are in the tertiary relationship.

9. Explain briefly why a secondary index cannot be sparse.

A secondary index must contain pointers to each record at the leaf level (per definition). It is thus dense.

10. Give an example of well-formed XML document (with at least two elements and one attribute) and its associated DTD.

Document:

```
<music>
<band country="UK" name="Joy Division">
<start_date>1979</start_date>
<split_date>1980</split_date>
</band>
<band name="Sigur Ros" country="Iceland">
<start_date>1994</start_date>
<split_date>Now</split_date>
</band>
</music>
```

DTD:

```
<!ELEMENT music(band*)>
<!ELEMENT band(start_date, split_date)>
<!ELEMENT start_date (#PCDATA)>
<!ELEMENT split_date (#PCDATA)>
<!ATTLIST band name CDATA required>
<!ATTLIST band country CDATA required>
```

Part 2 – Modelling, Relational Query Languages, XML (60%)

Consider the following relational schema (keys are underlined – note that the keys given are not realistic, they are defined this way to make your life simpler):

band(name, genre, country, start_date, split_date)

A band who plays music is defined by their name, they play a given genre of music (e.g., pop, rock). The band comes from a given country. The band was formed at start_date, and stopped playing together at the given split_date. When a band is still active the split_date is set to NULL.

musician(m_id, first_name, last_name, age, band_name, country)

A musician is identified by an id (m_id). A musician has a first name, a last name and an

age. Each musician is associated with a band.

festival(place, year, band_name)

A festival is a yearly concert where many bands appear. A festival is located in a given place (e.g., roskilde), and each year attracts many different bands.

Question 1 (5%)

Give the UML diagram of a conceptual model that corresponds to this relational schema.

Question 2 (15%)

1. I accept any answer as long as the explanation is ok. You might argue that yes, the relations are in BCNF because the only Fds are the ones where the key is on the left hand side. You might also argue that band_name -> musician is a valid FD, in which case the relation musician is not in BCNF.
2. The schema as it is given allows a musician to be part of one band only. How would you modify your UML diagram and the relational schema to allow (a) a musician to be part of at most 4 bands, and to (b) record the period during which a musician is in a band (e.g., Lol Tolhurst was with "The Cure" from 1976 to 1989 – currently "The Cure" has 4 members and 8 past members – note also that Simon Gallup has been with the cure in two periods from 1979-82 and 1985-present).

In terms of relational schema, we remove band from musician and create a new relation set-up :

musician(m_id, first_name, last_name, age, country)

set_up(mid, band_name, start_date, stop_date)

3. Write in SQL DDL a constraint that require that a festival schedules at least 4 bands. In practice how can this constraint be enforced?

Question 3 (5%)

Express the following queries in danish or english (all queries are expressed on the schema given in this text, i.e., a musician belongs to a single band):

1. SELECT band_name FROM festival WHERE place = "roskilde" and year = "2003";
Find the bands who played Roskilde in 2003

2. SELECT count(*) FROM musician m1, musician m2
WHERE m1.last_name = m2.last_name AND m1.m_id != m2.id;
Find the musicians with same last names.

3. SELECT m.first_name, m.last_name
FROM musician m1,
WHERE m.age = (SELECT avg(m2.age)
FROM musician m2 WHERE m2.country = m1.country);
Find the musicians whose age is the average musician age for their country.

Question 4 (25%)

Formulate the following queries in (a) relational algebra and (b) SQL (all queries should be expressed on the schema given in this text, i.e., a musician belongs to a single band):

1. Find the name (first and last) of the musicians that played in the band "The Smiths"
a) project first_name, last_name (select band_name = "The Smiths" musician)

b) SELECT first_name, last_name FROM musician WHERE band_name = "The Smiths";

2. Find the name (first and last) of all danish musicians playing in non-danish bands.

a) project first_name, last_name (select musician.country = "denmark" and musician.band_name = band.name and band.country != "denmark"
(musician x band))

b) SELECT first_name, last_name FROM musician m, band b
WHERE m.country = "denmark" and m.band_name = band.name
AND band.country != "denmark"

3. Find the pair of bands that have played in the same festival on the same year. Each pair should be listed only once (e.g., Mew, Nephew but not Nephew, Mew).

a) b1 = band
b2 = band
f1 = festival
f2 = festival

project b1.name, b2.name (select b1.name < b2.name and b1.name = f1.band_name and
b2.name = f2.band_name and f1.place = f2.place and f1.year = f2.year (f1 x b1 x f2 x b2)).

b) SELECT b1.name, b2.name FROM band b1, band b2, festival f1, festival f2
WHERE f1.band_name = b1.band and f2.band_name = b2.band and b1.name < b2.name
and f1.year = f2.year and f1.place = f2.place;

4. Find the name of the bands that played at all Roskilde festivals.

a) festival / project year, place (festival)

b) We use a helper tables that counts the number of instance of each festival (number of years at a given place) and number of participation of a band to a given festival in Roskilde:

```
SELECT band_name
FROM (SELECT place, count(year) yc
      FROM festival
      GROUP BY place) r,
(SELECT band_name, place, count(year) yc
 FROM festival
 GROUP BY band_name, place) s
WHERE r.place = s.place and r.place = "Roskilde" and r.yc = s.yc;
```

5. Find the name of the bands that played exactly twice at the Roskilde festival.

a) Exactly twice means "two or more" minus "three or more"

f1 = festival

f2 = festival

f3 = festival

two_or_more = project f1.band_name from (select f1.place = f2.place and f1.year != f2.year (f1xf2))

three_or_more = project f1.band_name from (select f1.place = f2.place and f1.place = f3.place f1.year != f2.year and f1.yaer != f3.year and
f2.year != f3.year (f1xf2xf3))

answer = two_or_more – three_or_more

b) Using aggregates, it is a bit simpler:

```
SELECT f.band_name
```

```
FROM festival f,
```

```
(SELECT band_name, place
```

```
FROM festival
```

```
GROUP BY band_name, place
```

```
HAVING count(year) = 2) r
```

WHERE f.band_name = r.band_name and r.place = "Roskilde";

Question 5 (10%)

Consider the following two bands: (Joy Division, alternative, UK, 1979, 1980) and (Sigur Ros, alternative, Iceland, 1994, NULL).

1. Give a well-formed XML document that contains both these tuples

Note that the choice of attributes/element is your own. There are many ways to do it. The important aspect is to have one single root for the document.

```
<music>
<band country="UK" name="Joy Division">
<start_date>1979</start_date>
<split_date>1980</split_date>
</band>
<band name="Sigur Ros" country="Iceland">
<start_date>1994</start_date>
<split_date>Now</split_date>
</band>
</music>
```

2. Using Xpath, write the following query "What is the name of all bands from the UK"?
//band[@country = "UK"]/@name